**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

# Лабораторна робота №1

з дисципліни

«Алгоритми та структури даних»

Виконав: студент групи ІМ-42

Максим Крамаренко Юрійович

номер варіанту: 17

Перевірив:

Сергієнко А. М.

Київ 2025

## Постановка задачі

Дане натуральне число n. Знайти суму перших n членів ряду чисел, заданого рекурентною формулою. Розв'язати задачу трьома способами:

1. У програмі використати рекурсивну функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному спуску;
2. У програмі використати рекурсивну функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному поверненні;
3. У програмі використати рекурсивну функцію, яка виконує обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні.

При проектуванні програм слід врахувати наступне:

- Програми повинні працювати коректно для довільного цілого додатного и включно з n = 1;
- Видимість змінних має обмежуватися тими ділянками, де вони потрібні;
- Функції повинні мати властивість модульності;
- У кожному з трьох способів рекурсивна функція має бути одна (за потреби, можна також використати додаткову функцію-обгортку (wrapper function));
- У другому способі можна використати запис (struct) з двома полями (але в інших способах у цьому немає потреби і це вважатиметься надлишковим);
- Програми мають бути написані мовою програмування C.

## Варіант 17:

Варіант № 17

$$F_1 = x; \quad F_{i+1} = F_i \cdot x^2/(4i^2 + 2i), \quad i > 0;$$

$$\sum_{i=1}^{n} F_i = \operatorname{sh} x, \quad |x| < 10^6.$$

## Текст програми:

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct{
   double F;
   double S;
} Result;

double r1(double x, int n, int i, double F, double S);
double r3(double x, int n, int i, double F);
Result r2(double x, int n, int i);
double loop(double x, int n, double S);

int main()
{
   double x = 1.5;
   int n = 5;
   int i = 1;
   double F, S;
   F = x;
   S = 0;

   printf("\n%*s\n", 25, "First recursion");
   printf("sh(%lf) = %lf\n", x, r1(x, n, i, F, S));

   printf("\n%*s\n", 25, "Second recursion");
   double res = r2(x, n, n).S;
   printf("sh(%lf) = %lf\n", x, res);

   printf("\n%*s\n", 25, "Third recursion");
   printf("sh(%lf) = %lf\n", x, r3(x, n, i, F));
```

```
    printf("\n%*s\n", 25, "Loop method");
    printf("sh(%lf) = %lf\n", x, loop(x, n, S));

    return 0;
}

double r1(double x, int n, int i, double F, double S) // Виконує обчислення n членів ряду, і суми на рекурсивному спуску
{
    if (-1000000 < x && x < 1000000)
    {
        S += F;
        printf("i = %d, F = %lf, S = %lf\n", i, F, S);
        if (n == 1)
        {
            return S;
        }
        else
        {
            F *= ((x*x) / (4*i*i + 2*i));
            return r1(x, n - 1, i + 1, F, S);
        }
    }
    else
    {
        printf("Error: x is out of range\n");
        return 0;
    }
}

Result r2(double x, int n, int i) // Виконує обчислення n членів ряду, і суми на рекурсивному поверненні
{
    if (-1000000 < x && x < 1000000)
    {
        Result res;
        if (i == 1)
        {
            res.F = x;
            res.S = x;
            printf("i = %d, F = %lf, S = %lf\n", i, res.F, res.S);
            return res;
        }
        Result prev = r2(x, n, i - 1);
```

```
        res.F = prev.F * (x*x) / (4*(i-1)*(i-1) + 2*(i-1));
        res.S = prev.S + res.F;
        printf("i = %d, F = %lf, S = %lf\n", i, res.F, res.S);
        return res;
    }
    else
    {
        printf("Error: x is out of range\n");
        return (Result){0, 0};
    }
}

double r3(double x, int n, int i, double F) // Виконує обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні
{
    if (-1000000 < x && x < 1000000)
    {
        double S;
        if (i == n)
        {
            S = F;
            printf("i = %d, F = %lf, S = %lf\n", i, F, S);
            return S;
        }
        S = r3(x, n, i + 1, F*(x*x) / (4*i*i + 2*i));
        S += F;
        printf("i = %d, F = %lf, S = %lf\n", i, F, S);
        return S;
    }
    else
    {
        printf("Error: x is out of range\n");
        return 0;
    }
}

double loop (double x, int n, double S) // Виконує обчислення n членів ряду, і суми на циклі
{
    if (-1000000 < x && x < 1000000)
    {
        double F = x;
        for (int i = 1; i <= n; i++)
        {
```

```
            S += F;
            printf("i = %d, F = %lf, S = %lf\n", i, F, S);
            F *= ((x*x) / (4*i*i + 2*i));
        }
        return S;
    }
    else
    {
        printf("Error: x is out of range\n");
        return 0;
    }
}
```

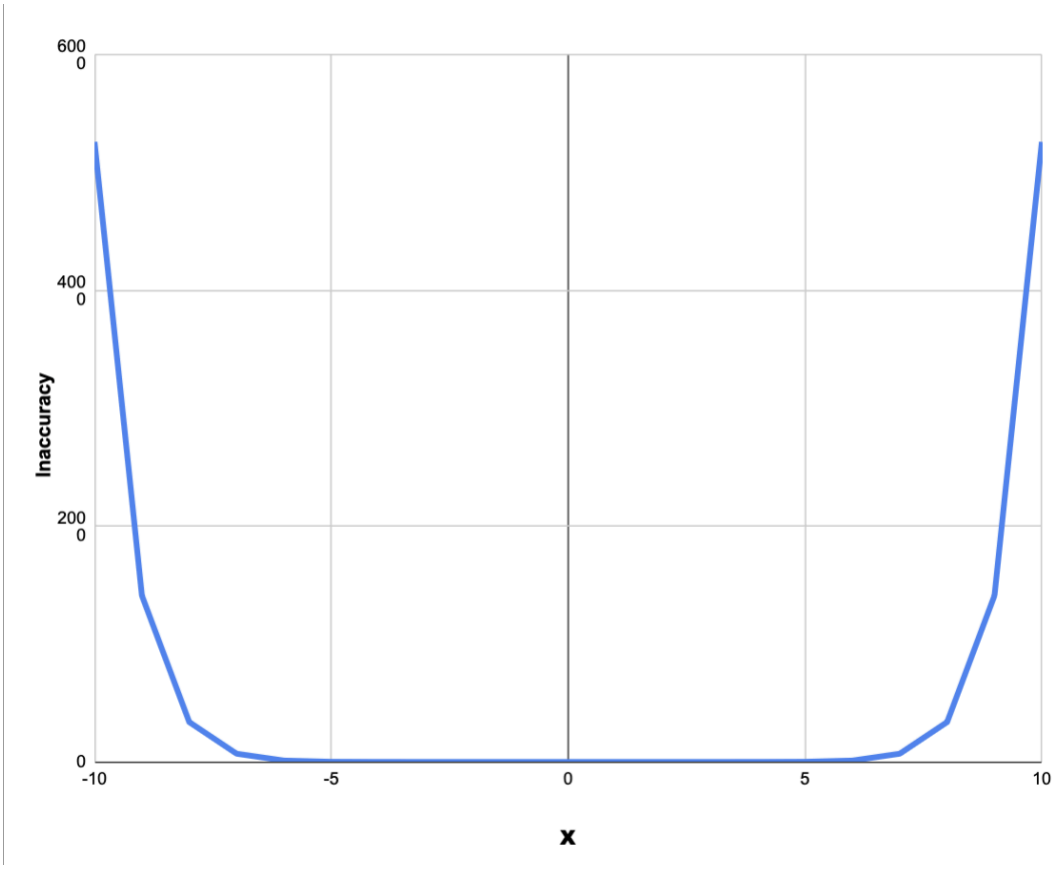
# Тестування програми:

```
maksymkramarenko@MacBook-Pro-Maksym Lab1 % ./prograr

          First recursion
 i = 1, F = 2.540000, S = 2.540000
 i = 2, F = 2.731177, S = 5.271177
 i = 3, F = 0.881023, S = 6.152201
 i = 4, F = 0.135334, S = 6.287534
 i = 5, F = 0.012127, S = 6.299661
 sh(2.540000) = 6.299661

         Second recursion
 i = 1, F = 2.540000, S = 2.540000
 i = 2, F = 2.731177, S = 5.271177
 i = 3, F = 0.881023, S = 6.152201
 i = 4, F = 0.135334, S = 6.287534
 i = 5, F = 0.012127, S = 6.299661
 sh(2.540000) = 6.299661

          Third recursion
 i = 5, F = 0.012127, S = 0.012127
 i = 4, F = 0.135334, S = 0.147460
 i = 3, F = 0.881023, S = 1.028483
 i = 2, F = 2.731177, S = 3.759661
 i = 1, F = 2.540000, S = 6.299661
 sh(2.540000) = 6.299661

             Loop method
 i = 1, F = 2.540000, S = 2.540000
 i = 2, F = 2.731177, S = 5.271177
 i = 3, F = 0.881023, S = 6.152201
 i = 4, F = 0.135334, S = 6.287534
 i = 5, F = 0.012127, S = 6.299661
 sh(2.540000) = 6.299661
```

```
maksymkramarenko@MacBook-Pro-Maksym Lab1 % ./program

          First recursion
 i = 1, F = 2.540000, S = 2.540000
 i = 2, F = 2.731177, S = 5.271177
 i = 3, F = 0.881023, S = 6.152201
 i = 4, F = 0.135334, S = 6.287534
 i = 5, F = 0.012127, S = 6.299661
 sh(2.540000) = 6.299661

         Second recursion
 i = 1, F = 2.540000, S = 2.540000
 i = 2, F = 2.731177, S = 5.271177
 i = 3, F = 0.881023, S = 6.152201
 i = 4, F = 0.135334, S = 6.287534
 i = 5, F = 0.012127, S = 6.299661
 sh(2.540000) = 6.299661

          Third recursion
 i = 5, F = 0.012127, S = 0.012127
 i = 4, F = 0.135334, S = 0.147460
 i = 3, F = 0.881023, S = 1.028483
 i = 2, F = 2.731177, S = 3.759661
 i = 1, F = 2.540000, S = 6.299661
 sh(2.540000) = 6.299661

             Loop method
 i = 1, F = 2.540000, S = 2.540000
 i = 2, F = 2.731177, S = 5.271177
 i = 3, F = 0.881023, S = 6.152201
 i = 4, F = 0.135334, S = 6.287534
 i = 5, F = 0.012127, S = 6.299661
 sh(2.540000) = 6.299661
```

```
maksymkramarenko@MacBook-Pro-Maksym Lab1 % ./program

          First recursion
 i = 1, F = -4.300000, S = -4.300000
 i = 2, F = -13.251167, S = -17.551167
 i = 3, F = -12.250704, S = -29.801870
 i = 4, F = -5.393226, S = -35.195097
 i = 5, F = -1.385011, S = -36.580107
 sh(-4.300000) = -36.580107

         Second recursion
 i = 1, F = -4.300000, S = -4.300000
 i = 2, F = -13.251167, S = -17.551167
 i = 3, F = -12.250704, S = -29.801870
 i = 4, F = -5.393226, S = -35.195097
 i = 5, F = -1.385011, S = -36.580107
 sh(-4.300000) = -36.580107

          Third recursion
 i = 5, F = -1.385011, S = -1.385011
 i = 4, F = -5.393226, S = -6.778237
 i = 3, F = -12.250704, S = -19.028940
 i = 2, F = -13.251167, S = -32.280107
 i = 1, F = -4.300000, S = -36.580107
 sh(-4.300000) = -36.580107

             Loop method
 i = 1, F = -4.300000, S = -4.300000
 i = 2, F = -13.251167, S = -17.551167
 i = 3, F = -12.250704, S = -29.801870
 i = 4, F = -5.393226, S = -35.195097
 i = 5, F = -1.385011, S = -36.580107
 sh(-4.300000) = -36.580107
```

**Графік залежності похибки обчислення заданої функції від значення x при фіксованому значенні n = 5**




## Висновок:

Засвоїв теоретичний матеріал лекцій набрався практичного досвіду в використовуванні рекурсивних алгоритмів і в написанні відповідних їм програм. Побачив, що похибка збільшується при збільшенні модуля x, але похибку можна зменшити взявши сумму більшої кількості членів ряда Тейлора(збільшити n).